I. AMENDMENTS TO THE CLAIMS

1. (currently amended) A computer implemented method for an analysis of ~~installing a plurality~~ of software components prior to deployment, comprising:

using a data structure in a storage that provides, for each of the plurality of software components, a component deployment dependency data, an indication of necessary components for an operation of each of the plurality of software components being installed, and an indication of incompatibility with a previously installed component,

using a computer connected to the storage and a program installed in a memory of the computer, performing the steps of:

determining a first plurality of components previously installed on the system;

determining a second plurality of components to be installed on the system;

accessing a third plurality of component deployment dependency data;

determining a fourth plurality of components suitable for parallel installation; and

determining an order in which the fourth plurality of components can be grouped for a fifth plurality of parallel installations;

accessing a sixth plurality of metadata from the data structure regarding the second plurality of components to be installed and accessing a seventh plurality of data regarding the first plurality of components previously installed; and

analyzing the sixth plurality of metadata to determine an eighth plurality of potential conflicts between the second plurality of components to be installed and the first plurality of components previously installed on the system;

wherein the pre-deployment analysis allows the second plurality of components to be installed in parallel and in a sequence of groups, whereby an installation time is reduced.

~~identifying components previously installed on a system;~~

~~identifying components to be installed on the system; and~~

~~identifying any potential conflicts between a previously installed component and a component to be installed.~~

2. Cancelled.

3. (currently amended) The method of claim ~~2~~1, further comprising updating the ~~semantic model~~ data structure with ~~the~~an identity of ~~newly~~ a ninth plurality of recently installed components.

4. (currently amended) The method of claim 1, further comprising providing a user with a plurality of options if ~~a conflict is identified~~for the eighth plurality of potential conflicts.

5. (currently amended) The method of claim 4, wherein a first option includes aborting ~~the~~ an installation.

6. (currently amended) The method of claim 4, wherein a second option includes continuing ~~the~~an installation.

7. (currently amended) The method of claim 6, further including, upon the exercise of the second option, recording an entry in a log indicative of ~~the~~a conflict and of ~~the~~a continuation of installation.

8. (currently amended) The method of claim 1, further comprising:

initiating a removal of a component from the system; and

identifying a tenth plurality of remaining components which depend on the component to be removed.

9. (currently amended) The method of claim 8, further comprising providing a user with a plurality of options if ~~a~~the tenth plurality of dependent remaining components is identified.

10. (currently amended) The method of claim 9, wherein a first option includes aborting ~~the~~a removal.

11. (currently amended) The method of claim 9, wherein a second option includes continuing ~~the~~a removal.

12. (currently amended) The method of claim 8, further comprising:

identifying a first component previously installed on the system which is dependent upon a removed component; and

determining ~~indicating the~~an identity of a second component upon which the first component depends.

13. (currently amended) The method of claim 12, further comprising:

installing the second component upon which the first component depends; and

creating a dependency link between the first component and the second components.

14. (currently amended) A system for an analysis of a plurality of software components to be conducted before installing the plurality of software components, comprising:

using a data structure in a storage that provides, for each of the plurality of software components, a component deployment dependency data, an identification of necessary components for an operation of each of the plurality of software components to be installed, and an identification of incompatibility with a previously installed component,

using a computer connected to the storage and a system, and a program installed in a memory of the computer,

means for determining a first plurality of components previously installed on the system;

means for determining a second plurality of components to be installed on the system;

means for accessing a third plurality of component deployment dependency data;

means for determining a fourth plurality of components suitable for parallel installation; and

means for determining an order in which the fourth plurality of components can be grouped for a fifth plurality of parallel installations;

means for accessing a sixth plurality of metadata from the data structure regarding the second plurality of components to be installed and accessing a seventh plurality of data regarding the first plurality of components previously installed; and

means for analyzing the sixth plurality of metadata to determine an eighth plurality of potential conflicts between the second plurality of components to be installed and the first plurality of components previously installed on the system;

wherein the pre-deployment analysis allows the second plurality of components to be installed in parallel and in a sequence of groups whereby an installation time is reduced.

~~means for identifying components previously installed on a system;~~

~~means for identifying components to be installed on the system; and~~

~~means for identifying any potential conflicts between a previously installed component and a component to be installed~~

15. Cancelled.

16. (currently amended) The system of claim 15, further comprising a means for loading an installation package including the~~semantic model~~data structure.

17. (currently amended) The system of claim 14, further comprising ~~a data structure comprising~~ a ninth plurality of references among the components to be installed and located in the data structure.

18. (currently amended) The system of claim 17, further comprising a means for accessing the data structure.

19. (currently amended) The system of claim 14, further comprising a means for installing the second plurality of components across a plurality of enterprise resources.

20. (currently amended) A data structure associated with a software component installation package adapted for execution on a computer, the data structure adapted for identifying a third plurality of potential conflicts between a second plurality of components to be installed on a

system and a first plurality of components previously installed on the system ~~component incompatabilities~~, comprising:

for each of the second plurality of software components, a component deployment dependency data, an indication of necessary components for an operation of each of the second plurality of software components, and an indication of incompatibility with one or more components of the first plurality of components ~~an entry for each component previously installed on a system~~; and

~~references associated with each previously component identifying any conflicting component to be installed;~~

~~whereby~~ wherein an alert is automatically generated if an attempt is made to install a component having an indication of incompatibility~~conflicting component~~.

21. (currently amended) A computer program product of a computer-readable medium usable with a programmable computer, the computer program product having computer-readable code embodied therein for pre-deployment analysis of ~~installing~~ software components, the computer-readable code comprising instructions for:

~~identifying components previously installed on a system;~~

~~identifying components to be installed on the system; and~~

~~identifying any potential conflicts between a previously installed component and a component to be installed.~~

determining a first plurality of components previously installed on the system;

determining a second plurality of components to be installed on the system;

accessing a third plurality of component deployment dependency data;

determining a fourth plurality of components suitable for parallel installation; and

determining an order in which the fourth plurality of components can be grouped for a fifth plurality of parallel installations;

accessing a sixth plurality of metadata from the data structure regarding the second plurality of components to be installed and accessing a seventh plurality of data regarding the first plurality of components previously installed; and analyzing the sixth plurality of metadata to determine an eighth plurality of potential conflicts between the second plurality of components to be installed and the first plurality of components previously installed on the system;

wherein the pre-deployment analysis allows the second plurality of components to be installed in parallel and in a sequence of groups, whereby an installation time is reduced.

22. Cancelled.

23. (Currently amended) The computer program product of claim 22, further comprising instructions for updating the ~~semantic model~~ data structure with ~~the~~ an identity of a ninth plurality of recently ~~newly~~ installed components.

24. (Original) The computer program product of claim 21, further comprising instructions for providing a user with a plurality of options if a conflict is identified.

25. (Currently amended) The computer program product of claim 24, wherein a first option includes aborting ~~the~~an installation.

26. (Currently amended) The computer program product of claim 24, wherein a second option includes continuing ~~the~~an installation.

27. (Currently amended) The computer program product of claim 26, further including instructions for, upon the exercise of the second option, recording an entry in a log indicative of the conflict and of ~~the~~a continuation of the installation.

28. (Currently amended) The computer program product of claim 21, further comprising instructions for:

initiating a removal of a component from the system; and

identifying a plurality of remaining components which depend on the component to be removed.

29. (Original) The computer program product of claim 28, further comprising instructions for providing a user with a plurality of options if a dependent remaining component is identified.

30. (Original) The computer program product of claim 29, wherein a first option includes aborting the removal.

31. (Original) The computer program product of claim 29, wherein a second option includes continuing the removal.

32. (Original) The computer program product of claim 28, further comprising instructions for:

identifying a first component previously installed on the system which is dependent upon a removed component; and

indicating the identity of a second component upon which the first component depends.

33. (Original) The computer program product of claim 32, further comprising instructions for:

installing the second component upon which the first component depends; and

creating a dependency link between the first and second components.

34. (Currently amended) A method for installing software components based upon a pre-installation analysis, comprising:

loading an installation package, the installation package including a ~~component compatibility~~ data structure;

searching a target to which components are to be installed to identify a plurality of previously installed components;

for a first component, accessing, in the ~~component compatibility~~ data structure, ~~the component~~ a component deployment dependency data, an indication of necessary components for an operation of the first component, and an indication of incompatibility with a previously installed component;

analyzing a plurality of data from the data structure to determine a plurality of conflicts between the first component to be installed and the plurality of components previously installed on the system~~compatibility data structure comprising, for each component A to be installed, a reference to any installed component with which the component A may conflict; and~~

~~determining whether a conflict is detected~~.

35. (Original) The method of claim 34, further comprising notifying a user of the conflict.

36. (Original) The method of claim 34, further comprising aborting the installation if a conflict is detected.

37. (Original) The method of claim 34, further comprising ignoring a detected conflict and continuing the installation.

38. (Original) The method of claim 37, further comprising entering a note in a log of the conflict.

39. (Currently amended) The method of claim 34, further comprising:

initiating the removal of an installed component;

accessing the ~~component compatability~~ data structure; and

identifying a conflict if the installed component is removed.

40. (Currently amended) The method of claim 34, further comprising:

initiating an installation of a second component ~~B~~;

searching a target to which the second component ~~B~~ is to be installed to identify installed components;

accessing the component ~~compatability~~ compatibility data structure; and

determining if all of the components required by the second component ~~B~~ are installed.

Please amend paragraph [26] as follows:

[26] The present invention employs a "semantic model" described more fully in commonly assigned and co-pending U.S. Patent Serial Number 10/727,011——, filed December 3, 2003——, ~~IBM Disclosure RSW8-2003-0414,~~ entitled e~~REGISTRY RECORDER AND ROLL BACK~~AUTONOMIC ROLLBACK, hereby incorporated by reference. Such a model, generated by the developer and included in the installation package, provides a "taxonomy" of all software components of interest, such as all software which IBM, for example, produces or uses. The model comprises a set of entries for each application, component and sub-component being installed (hereinafter collectively referred to as "components"). The model includes:

references or links among components indicating their deployment dependencies;

entries indicating what other components are necessary for the proper operation of each component being installed; and

entries indicating incompatibilities with other components likely to have been previously installed.

Please amend paragraph [28] as follows:

[28] The semantic model is a data structure stored in a knowledge base (as more fully described in commonly-assigned and co-pending U.S. Patent Serial Number——10/726,192, filed———December 2, 2003, ~~IBM Disclosure Number RSW8-2003-0413,~~ entitled HOSTING ENVIRONMENT ABSTRACTION AGENTS, hereby incorporated by reference). The data structure need not be any particular structure; examples of possible structures include (but are not limited to) a flat file, a database, an object model, etc. The component semantic model is generated by the developer and may be bundled with the deployment package or accessed from a remote site during installation. In the event that deployment is to occur across domains, the model may be augmented with a list of target machines on which components will be installed.

Please amend paragraph [33] as follows:

[33] The present invention also identifies potential component conflicts by implementing a pre-deployment hosting environment analyzer. Again the semantic model for software components is employed which captures the topology of software components at different levels of detail as well as capturing complex relationships among components. The deployed components on the target are recorded in the eRegistry. The installation is as follows: as soon as an eReadme file is available to deploy (an eReadme captures the information about the components that are to be deployed), the eRegistry is examined and the knowledge base (as more fully described in commonly-assigned and co-pending U.S. Patent Serial Number 10/725,612, filed December 2, 2003, entitled OPTIMAL COMPONENT INSTALLATION) is accessed to download metadata about the relationship among the components to be installed and the components existing in the target. Next, the relationship data is analyzed so appropriate action may be taken in the event that a conflict is identified. For example, the installation may continue or the user may be alerted of the possible conflict. In the event installation continues, an entry may be recorded in a log for later reference. As soon the software is deployed on the target, the target eRegistry is updated with appropriate installation information.

Respectfully submitted,

Rudolf O. Siegesmund

Rudolf O. Siegesmund
Registration No. 37,720
Gordon & Rees LLP
Suite 2650
2100 Ross Avenue
Dallas, Texas 75201
214-231-4703
214-461-4053 (fax)
rsiegesmund@gordonrees.com